Pinned Middleware Services on Many-core Platform for Exascale Systems

1. Challenge Addressed

Future exascale systems will be built on top of multi-core/many-core hardware architectures. Many studies try to explore techniques to improve the performance of such multi-core and many-core processors from both hardware and software sides. On hardware side, a concept of heterogeneous processors introduces hardware accelerator which chooses the most efficient hardware architecture to execute a specific type of workload [11]. However, the complexity of hardware compatibility is always an obstacle to heterogeneous processors. On software side, traditional operating systems scale well within a small number of cores on multi-core processors but not on many-core processors [3,13], would be even more difficult for future exascale systems.

A software concept to resolve design complexity to adapt hardware heterogeneity is widely used via a software layer called "middleware". It defines a common program abstraction that does not depend on hardware architecture but the operating system [12]. It is a very innovative design that globally standardizes programming language by its portable capability to write once and execute anywhere. It is also a perfect layer for setting up Quality-of-Service (QoS) guarantee for user applications since it performs program abstraction management between user applications and operating system. However, the middleware on traditional operating system is bloated with considerable overhead, which slows down user applications. This is the impact of resource being shared by both user applications and middleware on a traditional operating system.

Subsequently, middleware services have been deployed onto modern mutli-/many-core-based computing systems and data centers. Power and energy consumption is of critical concern for such systems, as well as future exascale systems. The hardware supported frequency scaling and voltage scaling are two popular techniques that have been generally used and studied nowadays. A coherent hardware and software design must be stated to ensure that the software design incorporates well with the hardware architecture to adapt its own frequency and voltage levels to maintain the system performance while keeping the power consumption as low as possible, in an effort to optimize overall energy consumption.

Finally, the increasing number of cores, shared memory access, execution parallelization, operating system scalability, middleware overhead and energy awareness are now inevitable issues to aggrandize the performance and efficiency of multi-core and many-core processors technology. However, there is no research addressing the need to coherently collaborate these aggravating issues together with bloated middleware services. As a consequence, we propose to pin the middleware services to mitigate the impact from their overhead and to evaluate the relation and effectiveness of pinned middleware services via an implementation on many-core processor platform with energy awareness for exascale systems.

2. Maturity

Lots of researchers are improving the operating system design to utilize hardware resources available on many-core systems since the performance of many-core systems cannot be fully achieved by traditional operating systems. At this moment, the design on many-core operating systems can be diverting into four different categories. The first category stands for time-shared operating system, which is actually the traditional operating system like Windows and Unix/Linux systems. The scheduling for resources and processes are done by a time-multiplexing function. The second category stands for space-shared operating system, which offers a totally different resource management to accommodate memory access pattern problems by using a resource scheduling that totally shares nothing between the cores. Corey [2] and fOS [3] are the representatives of the second category. The third category is called satellite operating system. It adds a top time-shared scheduling layer to manage the entire system with one central scheduler distributing processes to different hardware set which lets a local architecture manages its own execution. HeliOS [5] belongs to the third category. Barrelfish [4] and Tessellation [6] fall onto the last category, which is called library operating system. It adds a top space-shared scheduling layer as a resource allocator for local time-shared operating systems, which could be of any traditional operating system. Most of the many-core operating system designs address the same essential demand on a fast memory unit that supports fast inter-communication among the cores. Unfortunately, most of the stated operating system studies were not yet implemented on many-core microprocessor system. The researchers only suggest approaches to connect hardware and software frameworks together.

3. Novelty

One aspect we find that has been overlooked is the bloated middleware layer that lies on top of the operating system. A group of researchers [9, 10] have conducted extensive experiments on hardware techniques and algorithm optimization on middleware services. Even though they successfully accelerated the middleware services up to a certain degree, the speedup is not enough to miniaturize such tremendous overhead from the bloated middleware layer.

Therefore, we propose to accelerate the middleware execution such that the hardware acceleration and algorithm optimization are still compatible with system design. The proposed implementation is to pin the middleware execution on many-core systems by isolating middleware resource sharing from operating system and user applications. The distributed applications, middleware services and operating system services can be executed in parallel through a hardware and software cooperative design.

To efficiently map the hardware and software frameworks together, the awareness on power and energy consumption becomes another relevant constraint for multi-core and many-core microprocessor system as the number of core is increasing massively, especially for exascale systems. The power and energy efficiency aware operating system on many-core microprocessor has been studied in [8] but it has not been experimented. It is foreseeable that all the cores are not needed to be run at the same time, as ome of the cores can be turned off for power saving [14]. An execution can be run at an appropriate frequency and voltage levels to optimize power consumption of the entire system. Correspondingly, this study proposes to integrate the energy efficiency awareness, pinned middleware services and many-core platform in a holistic fashion for the future exascale system design.

4. Uniqueness

By implementing this hardware-software framework, this study aims to achieve a higher parallelization level by distributing the middleware services on many-core microprocessors for exascale systems. This design alleviates resource competition between application, middleware service and operating system service running on the same platform. Hence, the performance and efficiency of the system can be greatly improved. This study also intends to setup a demonstrative approach to analyze the computation performance and energy efficiency of the proposed design.

5. Effort

The ultimate goal is to setup a framework that executes middleware services on many-core platform efficiently for future exascale systems. The services on the many-core platform are classified according to its parallelization level [1] to be executed at optimized voltage and frequency. Communication, workload balance and power management between the cores are the keys to utilize the system. The utilization metrics will be evaluated in terms of execution time, power consumption, energy consumption, energy-delay product and hardware performance counter.

6. Applicability

The outcome is to implement energy-aware software/hardware assisted pinned middleware services on many-core platform for exascale systems. Three common middleware services are proposed to be implemented as separate study cases, being Document Object Model (DOM) Extensible Markup Language (XML) Parser, Garbage Collector (GC) and Just-in-Time (JIT) Compiler, respectively. An intensive speculation of each proposed middleware service on a many-core system is expected to conclude a balanced voltage and frequency configuration that optimizes computation performance and energy consumption. The analysis on middleware service program characteristics also expects an interpretation/detection of slow-down factors that are incurred by the bottlenecks of the middleware service. Once successful, it can be applied to more sophisticated OS services, such as file management, process management, etc.

7. Summary

This study aims to establish a scalable hardware-software framework to map the middleware services, operating system, and the application together on many-core platform that can efficiently utilize a massive number of cores available on the exascale system. This study invigorates the existing middleware service structure which has a significant tendency to be widely used among modern programmers across any hardware abstraction. This study benefits all hardware architects and programmers that share a flexible adhesive software layer between hardware and software entities with an innovative distributed hardware-software design.

References

[1] Krste A. et al, "The Landscape of Parallel Computing Research: A View from Berkeley," EECS technical report, December, 2006

[2] Silas B., Haibo C., Rong C., Yandong M., Frans K., Robert M., Aleksey P., Lex S., Ming W., Yuehua D., Yang Z., and Zheng Z., "Corey: An Operating System for Many Cores. In Proceedings of the 8th Symposium on Operating Systems Design and Implementation", San Diego, CA, December 2008.

[3] Wentzlaff, D. and Agarwal, A., 2009. "Factored operating systems (fos): the case for a scalable operating system for multicores", SIGOPS Oper. Syst. Rev., Vol.43, Issue 2, pp.76-85, April, 2009

[4] Baumann A., Barham P., Dagand P.E., Harris Y., Isaacs R., Peter S., Roscoe T., Schüpbach A., and Singhania A., "The Multikernel: A new OS architecture for scalable multicore systems". In Proceedings of the 22nd ACM Symposium on OS Principles, Big Sky, MT, USA, October 2009

[5] Nightingale, E. B. and Hodson, O. and McIlroy, R. and Hawblitzel, C. and Hunt, G, "Helios: heterogeneous multiprocessing with satellite kernels", SOSP '09, pp. 221-234, 2009

[6] Colmenares A. J. and Bird S. and Cook H. and Pearce P. and Zhu D. and Shalf J.and Hofmeyr S. and Asanovi K. and Kubiatowicz J., "Resource Management in the Tessellation Manycore OS", HotPar' 10, 2010

[7] Van Tol M. W. and Jesshope C. R., 2011. "An operating system strategy for general-purpose parallel computing on many-core architectures", Advances in Parallel Computing, Volume 20, "High Performance Computing: From Grids and Clouds to Exascale", IOS Press (2011), pp 157-18, 2011

[8] Vajda, A., "Many-core Virtualization and Operating Systems, Programming Many- Core Chips", Springer US, isbn 978-1-4419-9739-5, pages 127-152, 2011

[9] Tang J., Liu S., Gu Z., Li X., and Gaudiot J., "Hardware-Assisted Middleware: Acceleration of Garbage Collection Operations", Proceedings of the 21st IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP 2010), Rennes, France, July 7-9, 2010

[10] Tang, J, Liu, S., Gu Z., Liu C. and Gaudiot J., " Memory-Side Acceleration for XML Parsing", The 8th IFIP International Conference on Networking and Parallel Computing (NPC 2011), Changsha, Hunan, China, October 21-23, 2011

[11] Gschwind, M.; Hofstee, H.P.; Flachs, B.; Hopkin, M.; Watanabe, Y.; Yamazaki, T.; , "Synergistic Processing in Cell's Multicore Architecture," Micro, IEEE , vol.26, no.2, pp.10-24, March-April 2006

[12] P. Bernstein. "Middleware: A Model for Distributed System Services." Communications of the ACM, 39:2, pp. 86−98, February 1996,

[13] Silas B., Austin T. C., Yandong M., Aleksey P., Frans M. K., Robert M.. and Nickolai Z., "An analysis of Linux scalability to many cores", OSDI'10, pp. 1-8, 2010

[14] Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, and Doug Burger. "Dark silicon and the end of multicore scaling". In Proceedings of the 38th annual international symposium on Computer architecture (ISCA '11).